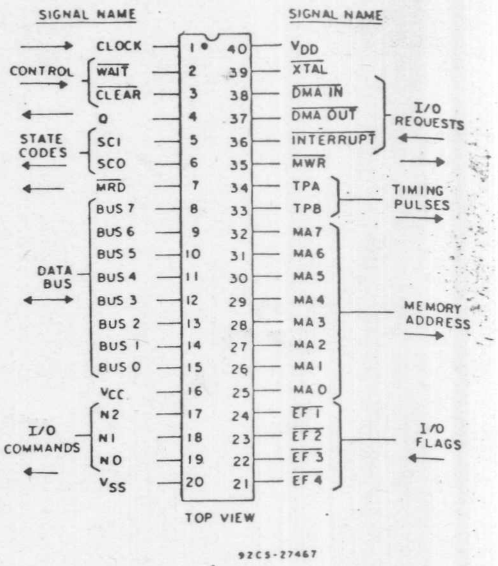
Preliminary CDP1802D, CDP1802CD

COSMAC Microprocessor

The RCA-CDP1802 is an LSI COS/MOS 8-bit register-oriented central-processing unit (CPU) designed for use as a general-purpose computing or control element in a wide range of stored-program systems or products.

It has the same basic COSMAC architecture as the CDP1801 microprocessor (see Fig. 5), but has an expanded instruction set, including a more powerful set of branch and ALU instructions, and enhanced hardware and performance features. The CDP1802 incorporates both the register and control chips of the CDP1801 on a single chip in a 40-lead hermetic dual-in-line ceramic package.

The CDP1802D is functionally identical to the CDP1802CD. The CDP1802D has a recommended operating voltage range of 3-12 volts; the CDP1802CD, a recommended operating voltage range of 4-6 volts.



Terminal Assignment for CDP1802

Features:

- Static silicon-gate CMOS circuitry—CD4000 series compatible
- Instruction fetch-execute time of 2.5/3.75 μs at V_{DD} = 10 V
- Compatible with CDP1801 software
- Full military-temperature range (−55 to +125°C)
- High noise immunity, wide operating-voltage range
- Single voltage supply
- No minimum clock frequency
- Low power
- TL compatible
- ngle-phase clock; optional on-chip crystal-controlled oscillator
- Simple control of reset, start, and pause
- 8-bit parallel organization with bidirectional data bus
- Any combination of standard RAM and ROM
- Memory addressing up to 65,536 bytes
- Flexible programmed I/O mode
- Program interrupt mode
- On-chip DMA
- Four I/O flag inputs directly tested by branch instructions
- Programmable output port
- 91 easy-to-use instructions
- 16 x 16 matrix of registers for use as multiple program counters, data pointers, or data registers

MAXIMUM RATINGS, *Absolute-Maximum Values*

Storage-Temperature Range (T_{stg}) .... −65 to +150°C

Operating-Temperature Range (T_A) .... −55 to +125°C

DC Supply-Voltage Range (V_{CC}, V_{DD})
(All voltage values referenced to V_{SS} terminal)
$V_{CC} \leqslant V_{DD}$:
CDP1802D .... −0.5 to +15 V
CDP1802CD .... −0.5 to +7 V

Power Dissipation Per Package (P_D):
For T_A = −55 to +100°C .... 500 mW
For T_A = +100 to +125°C .... Derate Linearly to 200 mW

Device Dissipation Per Output Transistor:
For T_A = −55°C to +125°C .... 100 mW

Input Voltage Range, All Inputs .... −0.5 to V_{DD} +0.5 V

Lead Temperature (During Soldering):
At distance 1/16 ± 1/32 inch (1.59 ± 0.79 mm) from case for 10 s max. .... +265°C

OPERATING CONDITIONS at T_A = 25°C Unless Otherwise Specified

For maximum reliability, nominal operating conditions should be selected so that operation is always within the following ranges.

CHARACTERISTIC	CONDITIONS		TYPICAL VALUES		UNITS
	V_{CC}[1] (V)	V_{DD} (V)	CDP1802D	CDP1802CD	
Supply-Voltage Range (At T_A = Full Package-Temperature Range)	–	–	3 to 12	4 to 6	V
Recommended Input Voltage Range	–	–	V_{SS} to V_{CC}	V_{SS} to V_{CC}	V
Clock Input Rise or Fall Time, t_r or t_f	3-15	3-15	5	5	μs
struction Time[2] (See Fig. 1)	5	5	5	5	μs
	5	10	3.2	–	
	10	10	2.5	–	
DMA Transfer Rate	5	5	400	400	KBytes/sec
	5	10	625	–	
	10	10	800	–	
Clock Input Frequency, f_{CL}	5	5	DC - 3.2	DC - 3.2	MHz
	5	10	DC - 5.0	–	
	10	10	DC - 6.4	–	
Clock Pulse Width, t_{WL}, t_{WH}	5	5	160	160	ns
	5	10	100	–	
	10	10	80	–	
Clear Pulse Width	5	5	300	300	ns
	5	10	200	–	
	10	10	150	–	

Notes:
1. $V_{CC} \leqslant V_{DD}$; for CDP1802CD V_{DD} = V_{CC} = 5 volts.
2. Equals 2 machine cycles—one Fetch and one Execute operation for all instructions except Long Branch and Long Skip, which require 3 machine cycles—one Fetch and two Execute operations.

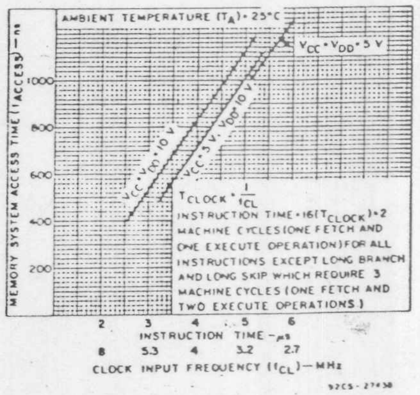


Fig. 1— Typical instruction time vs. memory system access time.

liminary CDP1802D, CDP1802CD

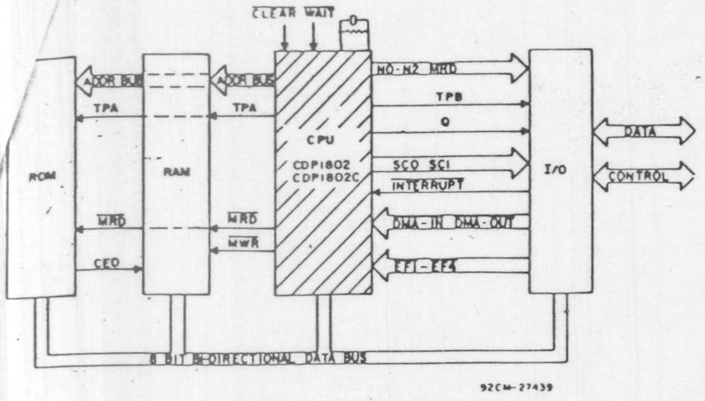


Fig. 2— Typical CDP1802 microprocessor system.

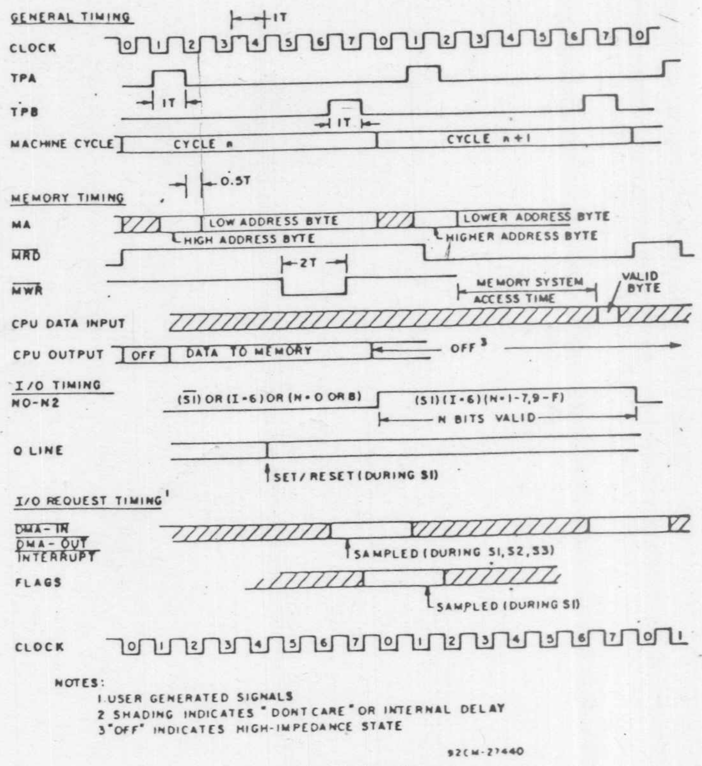


Fig. 3– Timing diagram.

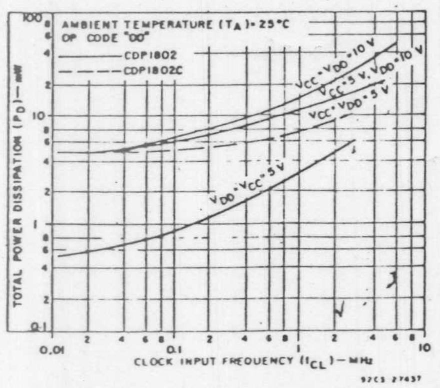


Fig. 4—Typical total power dissipation vs. clock input frequency.

ELECTRICAL CHARACTERISTICS at $T_A = 25^oC$

CHARACTERISTIC			CONDITIONS V_O (V)	V_{CC}, V_{DD} (V)	CDP1802D TYPICAL VALUES	CDP1802CD TYPICAL VALUES	UNITS
Quiescent Device Current, I_L			–	5, 5	100	500	µA
			–	10, 10	500	–	
			–	15, 15	1000	–	
Total Power Dissipation: OP CODE "00" (See F[illegible] 4)	f = MHz	3.2	–	5, 5	6	8	mW
		5.0	–	5, 10	30	–	
		6.4	–	10, 10	40	–	
Outpu[illegible]oltage: Low-Level, V_{OL}			–	5, 5	0.01	0.01	V
			–	10, 10	0.01	–	
High-Level, V_{OH}			–	5, 5	5	5	
			–	10, 10	10	–	
oise Immunity: Inputs Low, V_{NL}			0.5	5, 5	2.25	2.25	V
			−1	10, 10	3.45	–	
Inputs High, V_{NH}			4.5	5, 5	2.25	2.25	
			9	10, 10	3.45	–	
utput Drive Current: -Channel (Sink), I_DN			0.4	5, 5	1.5	1.5	mA
			0.5	10, 10	3.0	–	
-Channel (Source), I_DP			2.5	5, 5	−1.6	−1.6	
			4.6	5, 5	−0.4	−0.4	
			9.5	10, 10	−0.9	–	
put Leakage Current Any Input), I_{IL}, I_{IH}			–	5, 5	±1	±1	µA
			–	15, 15	±1	–	

ARCHITECTURE

The COSMAC block diagram is shown in Fig. 5. The principal feature of this system is a register array (R) consisting of sixteen 16-bit scratchpad registers. Individual registers in the array (R) are designated (selected) by a 4-bit binary code from one of the 4-bit registers labeled N, P, and X. The contents of any register can be directed to any one of the following three paths:

1. the external memory (multiplexed, higher-order byte first, on to 8 memory address lines);
2. the D register (either of the two bytes can be gated to D);
3. the increment/decrement circuit where it is increased or decreased by one and stored back in the selected 16-bit register.

The three paths, depending on the nature of the instruction, may operate independently or in various combinations in the same machine cycle.

With two exceptions, COSMAC instructions consist of two 8-clock-pulse machine cycles. The first cycle is the fetch cycle, and the second—and third, if necessary—are execute cycles. During the fetch cycle the four bits in the P designator select one of the 16 registers R(P) as the current program counter. The selected register R(P) contains the address of the memory location from which the instruction is to be fetched. When the instruction is read out from the memory, the higher-order 4 bits of the instruction byte are loaded into the I register and the lower-order 4 bits into the N register. The content of the program counter is automatically incremented by one so that R(P) is now "pointing" to the next byte in the memory.

reliminary CDP1802D, CDP1802CD

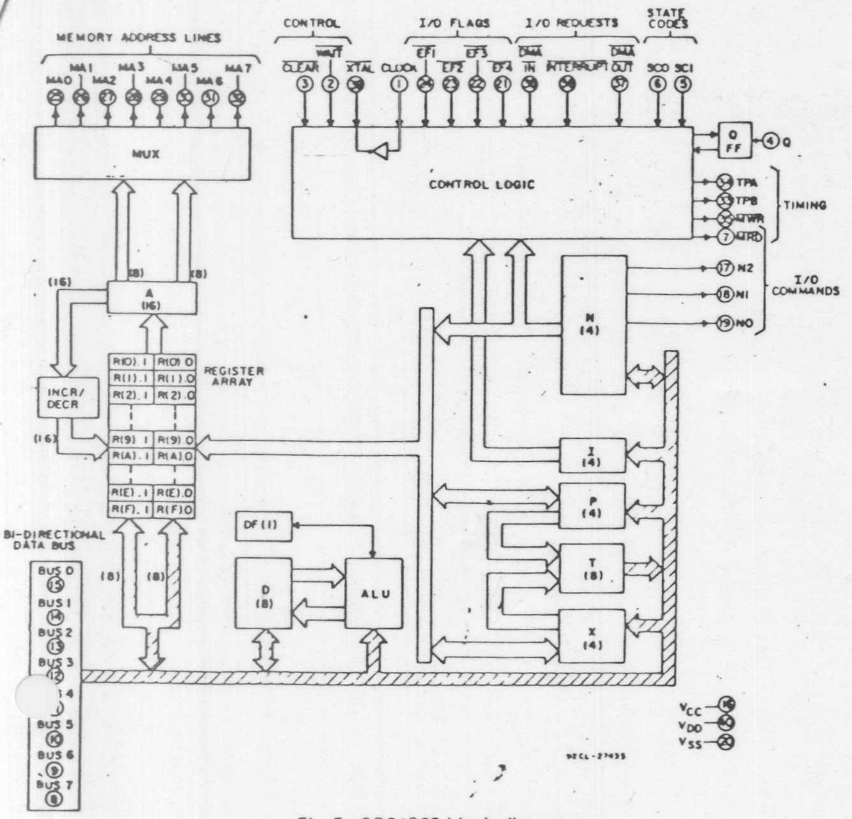


Fig. 5—CDP1802 block diagram.

The X designator selects one of the 16 registers R(X) to "point" to the memory for an operand (or data) in certain ALU or I/O operations.

The N designator can perform the following five functions depending on the type of instruction fetched:

1. designate one of the 16 registers in R to be acted upon during register operations;
2. indicate to the I/O devices a command code or device-selection code for peripherals;
3. indicate the specific operation to be executed during the ALU instructions, types of tests to be performed during the Branch instructions, or the specific operation required in a class of miscellaneous instructions (70-73 and 78-7B);
4. indicate the value to be loaded into P to designate a new register to be used as the program counter R(P);
5. indicate the value to be loaded into X to designate a new register to be used as data pointer R(X).

The registers in R can be assigned by a programmer in three different ways: as program counters, as data pointers, or as scratchpad locations (data registers) to hold two bytes of data.

Program Counters

Any register can be the main program counter; the address of the selected register is held in the P designator. Other registers in R can be used as subroutine program counters. By a single instruction the contents of the P register can be changed to effect a "call" to a subroutine. When interrupts are being serviced, register R(1) is used as the program counter for the interrupt servicing routine. At all other times the register designated as program counter is at the discretion of the user.

Data Pointers

The registers in R may be used as data pointers to indicate a location in memory. The register designated by X (i.e., R(X)) points to memory for the following instructions (see Table I):

1. ALU operations F0-F5, F7, 74, 75, 77;
2. output instructions 61 through 67;
3. input instructions 69 through 6F;
4. certain miscellaneous instructions—70-73, 78.

The register designated by N (i.e., R(N)) points to memory for the "load D from memory" instructions 0N and 4N and the "Store D" instruction 5N. The register designated by P (i.e., the program counter) is used as the data pointer for ALU instructions F8-FD, FF, 7C, 7D, 7F. During these instruction executions the operation is referred to as "data immediate".

Another important use of R as a data pointer supports the built-in Direct-Memory-Access (DMA) function. When a DMA-In or DMA-Out request is received, one machine cycle is "stolen". This operation occurs at the end of the execute machine cycle in the current instruction. Register R(0) is always used as the data pointer during the DMA operation. The data is read from (DMA-Out) or written into (DMA-In) the memory location pointed to by the R(0) register. At the end of the transfer, R(0) is incremented by one so that the processor is ready to act upon the next DMA byte transfer request. This feature in the COSMAC architecture saves a substantial amount of logic when fast exchanges of blocks of data are required, such as with magnetic discs or during CRT-display-refresh cycles.

A program load facility, using the DMA-In channel, is provided to enable users to load programs into the memory. This facility provides a simple, one-step means for initially entering programs into the microprocessor system and eliminates the requirement for specialized "bootstrap" ROM's.

Data Registers

When registers in R are used to store bytes of data, four instructions are provided which allow D to receive from or write into either the higher-order- or lower-order-byte portions of the register designated by N. By this mechanism (together with loading by data immediate) program pointer and data pointer designations are initialized. Also, this technique allows scratchpad registers in R to be used to hold general data. By employing increment or decrement instructions, such registers may be used as loop counters.

The Q Flip Flop

An internal flip flop, Q, can be set or reset by instruction and can be sensed by conditional branch instructions. The output of Q is also available as a microprocessor output.

Interrupt Servicing

Register R(1) is always used as the program counter whenever interrupt servicing is initiated. When an interrupt request comes in and the interrupt is allowed by the program (again, nothing takes place until the completion of the current instruction) the contents of the X and P registers are stored in the temporary register T, and X and P are set to new values; hex digit 2 in X and hex digit 1 in P. Interrupt enable is automatically deactivated to inhibit further interruptions. The interrupt routine is now in control; the contents of T are saved by means of a single instruction (78) in the memory location pointed to by R(X). At the conclusion of the interrupt, the routine restores the pre-interrupted values of X and P with a single instruction (70 or 71). The interrupt-enable flip-flop can be activated to permit further interrupts or can be disabled to prevent them.

minary CDP1802D, CDP1802CD

TABLE I – INSTRUCTION SUMMARY

(For Notes, see below)

INSTRUCTION	MNEMONIC	OP CODE	OPERATION
EMORY REFERENCE			
LOAD VIA N	LDN	0N	M(R(N))→D, FOR N NOT 0
LOAD ADVANCE	LDA	4N	M(R(N))→D, R(N) +1
LOAD VIA X	LDX	F0	M(R(X))→D
LOAD VIA X AND ADVANCE	LDXA	72	M(R(X))→D; R(X) +1
LOAD IMMEDIATE	LDI	F8	M(R(P))→D; R(P) +1
STORE VIA N	STR	5N	D→M(R(N))
STORE VIA X AND DECREMENT	STXD	73	D→M(R(X)); R(X) –1
REGISTER OPERATIONS			
INCREMENT REG N	INC	1N	R(N) +1
DECREMENT REG N	DEC	2N	R(N) –1
INCREMENT REG X	IRX	60	R(X) +1
GET LOW REG N	GLO	8N	R(N).0→D
PUT LOW REG N	PLO	AN	D→R(N).0
GET HIGH REG N	GHI	9N	R(N).1→D
PUT HIGH REG N	PHI	BN	D→R(N).1
LOGIC OPERATIONS♦♦			
OR	OR	F1	M(R(X)) OR D→D
OR IMMEDIATE,	ORI	F9	M(R(P)) OR D→D; R(P) +1
EXCLUSIVE OR	XOR	F3	M(R(X)) XOR D→D
EXCLUSIVE OR IMMEDIATE	XRI	FB	M(R(P)) XOR D→D, R(P) +1
AND	AND	F2	M(R(X)) AND D→D
AND IMMEDIATE	ANI	FA	M(R(P)) AND D→D; R(P) +1
SHIFT RIGHT	SHR	F6	SHIFT D RIGHT, LSB(D)→DF, 0→MSB(D)
SH[illegible] RIGHT WITH [illegible]RY	SHRC	76♦	SHIFT D RIGHT, LSB(D)→DF, DF→MSB(D)
RI[illegible] SHIFT RIGHT	RSHR		
SHIFT LEFT	SHL	FE	SHIFT D LEFT, MSB(D)→DF, 0→LSB(D)
SHIFT LEFT WITH CARRY	SHLC	7E♦	SHIFT D LEFT, MSB(D)→DF, DF→LSB(D)
RING SHIFT LEFT	RSHL		
ARITHMETIC OPERATIONS♦♦			
ADD	ADD	F4	M(R(X)) +D→DF, D
ADD IMMEDIATE	ADI	FC	M(R(P)) +D→DF, D; R(P) +1
ADD WITH CARRY	ADC	74	M(R(X)) +D +DF→DF, D
ADD WITH CARRY, IMMEDIATE	ADCI	7C	M(R(P)) +D +DF→DF, D R(P) +1
SUBTRACT D	SD	F5	M(R(X))–D→DF, D
SUBTRACT D IMMEDIATE	SDI	FD	M(R(P))–D→DF, D; R(P) +1
SUBTRACT D WITH BORROW	SDB	75	M(R(X))–D–(NOT DF)→DF, D
UBTRACT D WITH BORROW, IMMEDIATE	SDBI	7D	M(R(P))–D–(NOT DF)→DF, D; R(P) +1
UBTRACT MEMORY	SM	F7	D–M(R(X))→DF, D
UBTRACT MEMORY IMMEDIATE	SMI	FF	D–M(R(P))→DF, D; R(P) +1
UBTRACT MEMORY WITH BORROW	SMB	77	D–M(R(X))–(NOT DF)→DF, D
UBTRACT MEMORY WITH BORROW, IMMEDIATE	SMBI	7F	D–M(R(P))–(NOT DF)→DF, D R(P) +1
RANCH INSTRUCTIONS–SHORT BRANCH			
HORT BRANCH	BR	30	M(R(P))→R(P).0
O SHORT BRANCH (SEE SKP)	NBR	38♦	R(P) +1
HORT BRANCH IF D=0	BZ	32	IF D=0, M(R(P))→R(P).0 ELSE R(P) +1
[illegible] BRANCH IF D NOT 0	BNZ	3A	IF D NOT 0, M(R(P))→R(P).0 ELSE R(P) +1
HORT BRANCH IF DF=1	BDF	33♦	IF DF=1, M(R(P))→R(P).0 ELSE R(P) +1
HORT BRANCH IF POS OR ZERO	BPZ		
HORT BRANCH IF EQUAL OR GREATER	BGE		
HORT BRANCH IF DF=0	BNF	3B♦	IF DF=0, M(R(P))→R(P).0 ELSE R(P) +1
HORT BRANCH IF MINUS	BM		
HORT BRANCH IF LESS	BL		
HORT BRANCH IF Q=1	BQ	31	IF Q=1, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF Q=0	BNQ	39	IF Q=0, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF1=1	B1	34	IF EF1=1, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF1=0	BN1	3C	IF EF1=0, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF2=1	B2	35	IF EF2=1, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF2=0	BN2	3D	IF EF2=0, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF3=1	B3	36	IF EF3=1, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF3=0	BN3	3E	IF EF3=0, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF4=1	B4	37	IF EF4=1, M(R(P))→R(P).0 ELSE R(P) +1
ORT BRANCH IF EF4=0	BN4	3F	IF EF4=0, M(R(P))→R(P).0 ELSE R(P) +1

INSTRUCTION SUMMARY (CONT'D)

INSTRUCTION	MNEMONIC	OP CODE	OPERATION
BRANCH INSTRUCTIONS–LONG BRANCH			
LONG BRANCH	LBR	C0	M(R(P))→R(P).1 M(R(P) +1)→R(P).0
NO LONG BRANCH (SEE LSKP)	NLBR	C8♦	R(P) +2
LONG BRANCH IF D=0	LBZ	C2	IF D=0, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
LONG BRANCH IF D NOT 0	LBNZ	CA	IF D NOT 0, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
LONG BRANCH IF DF=1	LBDF	C3	IF DF=1, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
LONG BRANCH IF DF=0	LBNF	CB	IF DF=0, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
LONG BRANCH IF Q=1	LBQ	C1	IF Q=1, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
LONG BRANCH IF Q=0	LBNQ	C9	IF Q=0, M(R(P))→R(P).1 M(R(P) +1)→R(P).0 ELSE R(P) +2
SKIP INSTRUCTIONS			
SHORT SKIP (SEE NBR)	SKP	38♦	R(P) +1
LONG SKIP (SEE NLBR)	LSKP	C8♦	R(P) +2
LONG SKIP IF D=0	LSZ	CE	IF D=0, R(P) +2 ELSE CONTINUE
LONG SKIP IF D NOT 0	LSNZ	C6	IF D NOT 0, R(P) +2 ELSE CONTINUE
LONG SKIP IF DF=1	LSDF	CF	IF DF=1, R(P) +2 ELSE CONTINUE
LONG SKIP IF DF=0	LSNF	C7	IF DF=0, R(P) +2 ELSE CONTINUE
LONG SKIP IF Q=1	LSQ	CD	IF Q=1, R(P) +2 ELSE CONTINUE
LONG SKIP IF Q=0	LSNQ	C5	IF Q=0, R(P) +2 ELSE CONTINUE
LONG SKIP IF IE=1	LSIE	CC	IF IE=1, R(P) +2 ELSE CONTINUE
CONTROL INSTRUCTIONS			
IDLE	IDL	00=	WAIT FOR DMA OR INTERRUPT; M(R(0))→BUS
NO OPERATION	NOP	C4	CONTINUE
SET P	SEP	DN	N→P
SET X	SEX	EN	N→X
SET Q	SEQ	7B	1→Q
RESET Q	REQ	7A	0→Q
SAVE	SAV	78	T→M(R(X))
PUSH X,P TO STACK	MARK	79	(X,P)→T; (X,P)→M(R(2)) THEN P→X, R(2)–1
RETURN	RET	70	M(R(X))→(X,P); R(X) +1 1→IE
DISABLE	DIS	71	M(R(X))→(X,P); R(X) +1 0→IE

INPUT–OUTPUT BYTE TRANSFER			
OUTPUT 1	OUT 1	61	M(R(X))→BUS; R(X) +1; N LINES = 1
OUTPUT 2	OUT 2	62	M(R(X))→BUS; R(X) +1; N LINES = 2
OUTPUT 3	OUT 3	63	M(R(X))→BUS; R(X) +1; N LINES = 3
OUTPUT 4	OUT 4	64	M(R(X))→BUS; R(X) +1; N LINES = 4
OUTPUT 5	OUT 5	65	M(R(X))→BUS; R(X) +1; N LINES = 5
OUTPUT 6	OUT 6	66	M(R(X))→BUS; R(X) +1; N LINES = 6
OUTPUT 7	OUT 7	67	M(R(X))→BUS; R(X) +1; N LINES = 7
INPUT 1	INP 1	69	BUS→M(R(X)); BUS→D; N LINES = 1
INPUT 2	INP 2	6A	BUS→M(R(X)); BUS→D; N LINES = 2
INPUT 3	INP 3	6B	BUS→M(R(X)); BUS→D; N LINES = 3
INPUT 4	INP 4	6C	BUS→M(R(X)); BUS→D; N LINES = 4
INPUT 5	INP 5	6D	BUS→M(R(X)); BUS→D; N LINES = 5
INPUT 6	INP 6	6E	BUS→M(R(X)); BUS→D; N LINES = 6
INPUT 7	INP 7	6F	BUS→M(R(X)); BUS→D; N LINES = 7

♦NOTE: THIS INSTRUCTION IS ASSOCIATED WITH MORE THAN ONE MNEMONIC. EACH MNEMONIC IS INDIVIDUALLY LISTED.

♦♦NOTE: THE ARITHMETIC OPERATIONS AND THE SHIFT INSTRUCTIONS ARE THE ONLY INSTRUCTIONS THAT CAN ALTER THE DF.
AFTER AN ADD INSTRUCTION:
DF = 1 DENOTES A CARRY HAS OCCURRED
DF = 0 DENOTES A CARRY HAS NOT OCCURRED
AFTER A SUBTRACT INSTRUCTION:
DF = 1 DENOTES NO BORROW. D IS A TRUE POSITIVE NUMBER
DF = 0 DENOTES A BORROW. D IS TWO'S COMPLEMENT
THE SYNTAX "–(NOT DF)" DENOTES THE SUBTRACTION OF THE BORROW

=An idle instruction initiates a repeating S1 cycle. The processor will continue to idle until an I/O request ($\overline{\text{INTERRUPT}}$, $\overline{\text{DMA-IN}}$, or $\overline{\text{DMA-OUT}}$) is activated. When the request is acknowledged, the IDLE cycle is terminated and the I/O request is serviced, and then normal operation is resumed.

♦NOTE THIS INSTRUCTION IS ASSOCIATED WITH MORE THAN ONE MNEMONIC. EACH MNEMONIC IS INDIVIDUALLY LISTED.

0000 1001

[illegible]

0000 1000

0000 0 [illegible]

eliminary CDP1802D, CDP1802CD

1. Long-Branch, Long Skip and No Op instructions are the only instructions that require three cycles to complete (1 fetch + 2 execute).

   Long-Branch instructions are three bytes long. The first byte specifies the condition to be tested; and the second and third byte, the branching address.

   The long-branch instructions can:
   - a) Branch unconditionally
   - b) Test for D=0 or D≠0
   - c) Test for DF=0 or DF=1
   - d) Test for Q=0 or Q=1
   - e) effect an unconditional no branch

   If the tested condition is met, then branching takes place; the branching address bytes are loaded in the high-and-low-order bytes of the current program counter, respectively. This operation effects a branch to any memory location.

   If the tested condition is not met, the branching address bytes are skipped over, and the next instruction in sequence is fetched and executed. This operation is taken for the case of unconditional no branch.

2. The short-branch instructions are two bytes long. The first byte specifies the condition to be tested, and the second specifies the branching address.

   The short-branch instructions can:
   - a) Branch unconditionally
   - b) Test for D=0 or D≠0
   - c) Test for DF=0 or DF=1
   - d) Test for Q=0 or Q=1
   - e) Test the status (1 or 0) of the four EF flags
   - f) Effect an unconditional no branch

   If the tested condition is met, then branching takes place; the branching address byte is loaded into the low-order byte position of the current program counter. This effects a branch with the current 256-byte page of the memory, i.e., the page which holds the branching address. If the tested condition is not met, the branching address byte is skipped over, and the next instruction in sequence is fetched and executed. This same action is taken in the case of unconditional no branch.

3. The skip instructions are one byte long. There is one Unconditional Short-Skip (SKP) and eight Long-Skip instructions.

   The Unconditional Short-Skip instruction takes 2 cycles to complete (1 fetch + 1 execute). Its action is to skip over the byte following it. Then the next instruction in sequence is fetched and executed. This SKP instruction is identical to the unconditional no-branch instruction (NBR) except that the skipped-over byte is not considered part of the program.

   The Long-Skip instructions take three cycles to complete (1 fetch + 2 execute).

   They can:
   - a) Skip unconditionally
   - b) Test for D=0 or D≠0
   - c) Test for DF=0 or DF=1
   - d) Test for Q=0 or Q=1
   - e) Test for IE=1

   If the tested condition is met, then Long Skip takes place; the current program counter is incremented twice. Thus two bytes are skipped over and the next instruction in sequence is fetched and executed. If the tested condition is not met, then no action is taken. Execution is continued by fetching the next instruction in sequence.

COSMAC Register Summary

D	8 Bits	Data Register (Accumulator)	N	4 Bits	Holds Low-Order Instr. Digit
DF	1 Bit	Data Flag (ALU Carry)	I	4 Bits	Holds High-Order Instr. Digit
R	16 Bits	1 of 16 Scratchpad Registers	T	8 Bits	Holds old X, P after Interrupt (X is high byte)
P	4 Bits	Designates which register is Program Counter	IE	1 Bit	Interrupt Enable
X	4 Bits	Designates which register is Data Pointer	Q	1 Bit	Output Flip Flop

INSTRUCTION SET

The COSMAC instruction summary is given in Table I. Hexadecimal notation is used to refer to the 4-bit binary codes.

In all registers bits are numbered from the least significant bit (LSB) to the most significant bit (MSB) starting with 0.

R(W): Register designated by W, where
W=N or X, or P

R(W).0: Lower-order byte of R(W)
R(W).1: Higher-order byte of R(W)
N0 = Least significant Bit of N Register

Operation Notation
$M(R(N)) \rightarrow D;\ R(N) + 1$

This notation means: The memory byte pointed to by R(N) is loaded into D, and R(N) is incremented by 1.

SIGNAL DESCRIPTIONS

BUS 0 to BUS 7 (Data Bus)	8-bit bi-directional DATA BUS lines. These lines are used for transferring data between the memory, the microprocessor, and I/O devices.
N0 to N2 (I/O Command)	Issued by an I/O instruction to signal the I/O control logic of a data transfer between memory and I/O interface. These lines can be used to issue command codes or device selection codes to the I/O devices (independently or combined with the memory byte on the data bus when an I/O instruction is being executed). The N bits are low at all times except when an I/O instruction is being executed. During this time their state is the same as the corresponding bits in the N register. The direction of data flow is defined in the I/O instruction by bit N3 and is indicated by the level of the MRD signal. $\overline{MRD} = V_{CC}$: Data from I/O to CPU and Memory $\overline{MRD} = V_{SS}$: Data from Memory to I/O
$\overline{EF1}$ to $\overline{EF4}$ (4 Flags)	These levels enable the I/O controllers to transfer status information to the processor. The levels can be tested by the conditional branch instructions. They can be used in conjunction with the INTERRUPT request line to establish interrupt priorities. These flags can also be used by I/O devices to "call the attention" of the processor, in which case the program must routinely test the status of these flag(s). The flag(s) are sampled at the beginning of every S1 cycle.

SIGNAL DESCRIPTIONS (Cont'd)

$\overline{\text{INTERRUPT}}$, $\overline{\text{DMA-IN}}$, $\overline{\text{DMA-OUT}}$ (3 I/O Requests)

These signals are sampled by the CDP1802 during the interval between the leading edge of TPB and the leading edge of TPA.

Interrupt Action: X and P are stored in T after executing current instruction; designator X is set to 2; designator P is set to 1; interrupt enable is reset to 0 (inhibit); and instruction execution is resumed.

DMA Action: Finish executing current instruction; R(0) points to memory area for data transfer; data is loaded into or read out of memory; and increment R(0).

Note: In the event of concurrent DMA and INTERRUPT requests, DMA-IN has priority followed by DMA-OUT and then INTERRUPT.

SC0, SC1, (2 State Code Lines)

These lines indicate that the CPU is: 1) fetching an instruction, or 2) executing an instruction, or 3) processing a DMA request, or 4) acknowledging an interrupt request. The levels of state code are tabulated below. All states are valid at TPA. H = V_{CC}, L = V_{SS}.

State Type	State Code Lines	
	SC1	SC0
S0 (Fetch)	L	L
S1 (Execute)	L	H
S2 (DMA)	H	L
S3 (Interrupt)	H	H

TPA, TPB (2 Timing Pulses)

Positive pulses that occur once in each machine cycle (TPB follows TPA). They are used by I/O controllers to interpret codes and to time interaction with the data bus. The trailing edge of TPA is used by the memory system to latch the higher-order byte of the 16-bit memory address. TPA is suppressed in IDLE when the CPU is in the load mode.

MA0 to MA7 (8 Memory Address Lines)

The higher-order byte of a 16-bit COSMAC memory address appears on the memory address lines MA0-7 first. Those bits required by the memory system are strobed into external address latches by timing pulse TPA. The low-order byte of the 16-bit address appears on the address lines after the termination of TPA. Latching of all 8 higher-order address bits would permit a memory system of 64K bytes.

$\overline{\text{MWR}}$ (Write Pulse)

A negative pulse appearing in a memory-write cycle, after the address lines have stabilized.

$\overline{\text{MRD}}$ (Read Level)

A low level on $\overline{\text{MRD}}$ indicates a memory read cycle. It can be used to control three-state outputs from the addressed memory which may have a common data input and output bus. If a memory does not have a three-state high-impedance output, $\overline{\text{MRD}}$ is useful for driving memory/bus separator gates. It is also used to indicate the direction of data transfer during an I/O instruction:

$\overline{\text{MRD}}$ = V_{CC}: Data from I/O to CPU and Memory

$\overline{\text{MRD}}$ = V_{SS}: Data from Memory to I/O

Q

Single bit output from the CPU which can be set or reset under program control. During SEQ or REQ instruction execution, Q is set or reset between the trailing edge of TPA and the leading edge of TPB.

CLOCK

Input for externally generated single-phase clock. A typical clock frequency is 6.4 MHz at $V_{CC} = V_{DD}$ = 10 volts. The clock is counted down internally to 8 clock pulses per machine cycle.

$\overline{\text{XTAL}}$

Connection to be used with clock input terminal, for an external crystal, if the on-chip oscillator is utilized. The crystal is connected between terminals 1 and 39 (CLOCK and $\overline{\text{XTAL}}$) in parallel with a resistance (10 megohms typ.). Frequency trimming capacitors may be required at terminals 1 and 39.

liminary CDP1802D, CDP1802CD

SIGNAL DESCRIPTION (Cont'd)

$\overline{\text{WAIT}}$, $\overline{\text{CLEAR}}$
(2 Control Lines)

Provide four control modes as listed in the following truth table:

$\overline{\text{CLEAR}}$	$\overline{\text{WAIT}}$	MODE
L	L	Load
L	H	Reset
H	L	Pause
H	H	Run

The function of the modes are defined as follows:

Load

Holds the CPU in the IDLE execution state and allows an I/O device to load the memory without the need for a "bootstrap" loader. It modifies the IDLE condition so that DMA-IN operation does not force execution of the next instruction.

Reset

Registers I, N, Q are reset, IE is set and 0's (V_{SS}) are placed on the data bus. TPA and TPB are suppressed while reset is held and the CPU is placed in S1. The first machine cycle after termination of reset is an initialization cycle. During this cycle the CPU remains in S1 and registers X, P, and R(0) are reset. Interrupt and DMA servicing are suppressed during the initialization cycle.

The next cycle is an S0, S1, or an S2 but never an S3. With the use of a 71 instruction followed by 00 at memory locations 0000 and 0001, this feature may be used to reset IE, so as to preclude interrupts until ready for them. Power-up reset can be realized by connecting an external RC to $\overline{\text{CLEAR}}$.

Pause

Stops the internal CPU timing generator on the first negative high-to-low transition of the input clock. The oscillator continues to operate, but subsequent clock transitions are ignored.

Run

May be initiated from the Pause or Reset mode functions. If initiated from Pause, the CPU resumes operation on the first negative high-to-low transition of the input clock. When initiated from the Reset operation, the first machine cycle following Reset is always the initialization cycle. The initialization cycle is then followed by a DMA (S2) cycle or fetch (S0) from location 0000 in memory.

V_{DD}, V_{SS}, V_{CC}
(Power Levels)

The internal voltage supply V_{DD} is isolated from the Input/Output voltage supply V_{CC} so that the processor may operate at maximum speed while interfacing with various external circuit technologies, including T^2L at 5 volts. V_{CC} must be less than or equal to V_{DD}. All outputs swing from V_{SS} to V_{CC}. The recommended input voltage swing is V_{SS} to V_{CC}.

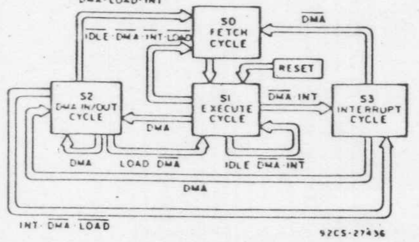


Fig. 6—CDP1802 microprocessor state transitions (Run Mode).

The CDP1802 and CDP1802C CPU state transitions when in the RUN mode are shown in Fig. 6. Each machine cycle requires the same period of time—8 clock pulses. The execution of an instruction requires either two or three machine cycles, S0 followed by a single S1 cycle or two S1 cycles. S2 is the response to a DMA request and S3 is the interrupt response.